MVA2000 IAPR Workshop on Machine Vision Applications, Nov. 28–30, 2000, The University of Tokyo, Japan

3—35

An Environment to Test Progressive Refinement of Indexing for Content-Based Image Retrieval

Maria Grazia Albanesi, Marco Ferretti, Alessandro Giancane
University of Pavia, Dipartimento di Informatica e Sistemistica
Via Ferrata n. 1 – 27100 Pavia, ITALY
Email: {albanesi, ferretti, gianca}@elzira.unipv.it

Abstract

Content-based image retrieval is a fairly new discipline. Yet research in this field has highlighted many approaches that show good performance in specific sub-problems using single filtering criteria, such as color based histograms, shape description or texture analysis.

Most systems built so far use multiple access methods although they compose them according to a pre-defined strategy. In this paper we report an environment that allows the evaluation of different type of filtering composition strategies. The aim is to let the developers work with a quick tool for fine tuning of sequences or compositions of methods, especially when such methods are based on different features. The selectivity of each indexing method is tested against a single database of images and then the overall performance of a sequence or a composition of methods is quickly estimated. The experiments have been applied on filtering criteria that share a multiresolution approach, in order to highlight the role that the hierarchical indexing approach can play in reducing the overall computational cost.

1 Introduction

In this work we present an environment developed to integrate, test and calibrate efficient multiple search strategies for content based image retrieval. We assume in the following that the search space consists of a very large set of images, that can be collected into sub-sets sharing some characteristic. We explicitly avoid making any a-priori assumption on the sub-sets, rather we postulate that any content-based query can draw from single or a set of indexing methods.

Indeed, existing approaches for multiple search queries usually combine a few classes of feature based methods, such as color, shape, and texture. Among these systems there are QBIC [1], PhotoBook [2], Virage [3] and VisualSEEk [4].

Rather than a single tool that embeds a customized feature based indexing scheme, we have preferred a more detailed approach. We anticipate that there are a plenty of different methods within each common approach (color, shape, texture and other); for example, a few color based methods could be based on histogram intersection or hierarchical histogram computations or other different approaches. The rationale behind this assumption is that a huge unconstrained database of images must draw from quite different indexing methods and that a variation of each method allows a more accurate tuning against different image semantics.

In content based image retrieval, user feedback relevance is a sound approach in most cases. This is especially true when:

- the user is an expert and he/she knows well how the methods could highlight the image features;
- the methods available are just a few and the database is extremely heterogeneous.

On the contrary, when the user has no specific knowledge of the indexing methods, when the database is huge and consists of collections of fairly similar images or when a large set of methods are available for querying, user feedback can hardly offer a sound hint to query refinement. In these situations a more quantitative approach becomes mandatory.

Quantitative indexing in traditional DBMS heavily uses indexes statistics for exploiting the selectivity. We pursue this direction in image databases indexing. This approach requires a tool a) to build the statistics (such as clustering); b) to build a correct and optimized querying strategy.

In this contribution we will not discuss clusterization techniques but we will focus on querying strategy optimization; we exploit the indexing methods composition to highlight a more accurate selectivity. We have chosen to analyze indexing methods based on hierarchically defined features; the hierarchical structure allows the reduction of the computational complexity both in the index set up phase and also in the index accessing phase [5].

We have designed and implemented a framework to develop, test and calibrate indexing strategies; a fairly easy environment that supports: i) the introduction of new indexing methods; ii) the evaluation of each method against a newly collected image database; iii) relative comparison of different methods against a common database; iv) the evaluation of different strategies to highlight the optimum sequence for a specific application.

2 System architecture and design issues

This section describes the overall systems. We begin with a set of definitions to introduce the terminology used in the following and to describe the assumptions we make on the construction of indexing methods. The next section describes the architecture that implements the whole search engine and its software implementation.

2.1 Preliminary definitions

We define an *indexing method* as a procedure that processes an image and returns a *signature*. The signature, according to a specified method version, consists of a given length vector of coefficients. The signature embeds a hierarchical structure, if the corresponding method is a hierarchical one. The collection of signatures, computed using a common method, make up the *index*.

Although the image format is method independent, we choice to collect into a homogeneous format (BMP icons, scaled to a proper dimension); we call these collections *image catalogs*. Currently we resize all images to 128x128 pixels before process them with an *indexing method*.

Methods are applied to catalogs, thus to build indexes for each catalog. Obviously a catalog could be processed by more methods, so it could be indexed and browsed by a variety of admissible indexing criteria.

Queries are expressed against a chosen catalog. We distinguish among: i) single image queries (*Interactive Queries*); ii) image set queries (*Table Queries*); ii) whole catalog queries (*Catalog Queries*). Queries consist of processing the search images with a single or a composition of methods among those available. In the following we will refer to *Table* and *Catalog Query* as *Batch Queries*.

The results of each query consist of the ordered target catalog; the ordering is obtained through ranking of signature distances measured with a chosen metric, currently L_2. The distances are computed between the signature of the search image (*pattern signatures*) and those stored in the indexes of the catalog.

Methods composition can be chosen between two kind of composition strategies: *parallel* or *serial composition*. *Parallel composition* consists of running multiple versions of the query using a set of methods against the same image catalog; returned signature distances are normalized at first, then we weight each returned set with relevance coefficients (specified by the user during the set up of the query) and we order the whole set of weighted results to obtain the final ordered set of images.

In *serial composition* (referred to as *sequence* in the following) there are n queries executed according to n methods: each query uses a single indexing method and each query output becomes the target set for the subsequent evaluation. The cardinality of the output set, that is the percent threshold of images that will be retained at each evaluation, is specified by the user during the set up of the query.

3 Architecture overview

This section deals with the description of our tool and how it has been designed to implement the requirements described in the previous sections. The key issues are the user friendly interface together with the possibility to easy integrate new indexing and analysis methodologies and the easy definition and management of querying strategies based on single or multiple methods composition. The choice to support different kinds of users (indexing methods developer and querying strategy analysts) within a common framework has required the design of a structure characterized by easy maintenance and easy extensibility possibilities.

A scheme showing the main components of our tool is depicted in **Figure 1**. The scheme highlights how the different modules are integrated into a common architecture. *Result Analysis*, *Image Processing* and *Indexing Methods* blocks are the three key modules that allow the two main uses of the system:

i) query by content on some predefined catalog;
ii) qualitative and quantitative result analysis of single or customized composition of indexing methods.

The three blocks (*Results Analysis, Image Processing, Indexing Methods*) communicate with their respectively external mathematical libraries using a specified interface (yellow layer, **Figure 1**) and store or read data, using a different interface (red layer, **Figure. 1**), through the *DB System Tools* block.

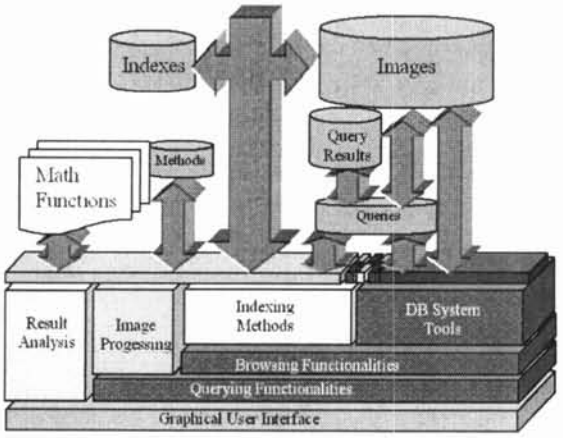

Figure 1. Architecture overview

Thanks to these two standardized interface layers, when a new indexing method has been developed and declared into the system, the users are enabled to apply all the functionalities available in the system on the new installed method allowing immediately its test and analysis.

Obviously, this framework requires the use of a strict communication protocol between the three blocks (*Result Analysis, Image Processing* and *Indexing Methods*) and their own mathematical functions (defined in external

modules). However this choice allows a strong reduction of the effort due the maintenance and extendibility of the external mathematical functions within the remaining application code.

A typical life cycle of a new indexing method for Image Database (*IDB*) consists of an iterated sequence of phases. Two main phases of this sequence are: i) the mathematical implementation of the method and ii) its evaluation on different sets of images. Obviously, these two phases are often iterated to allow a progressive refinement of the indexing algorithm. It is clear that the refinement process does not influence the way of dealing with the *IDB* by the whole of the application, but could be limited only to the set of functions that deal only with the index construction.

Moreover, since the core of a new indexing method is at first implemented externally in a different environment (we use MathWorks Matlab); maintaining the indexing algorithm details in external library, it is possible to integrate them within the rest of application by simple operations. This allows the users to move quickly from the indexing method algorithm implementation into the corresponding analysis, evaluation and customization phases.

The choice to use this structure allows also, an easier construction for those indexing methods based on the integration of different methods. A new method could be easy defined using two or more just defined methods; the user needs only the single method definitions with a set of parameters required for their interaction, without been involved with the proper algorithms of each method.

3.1 The software

An application, implementing the scheme described so far, has been developed exploiting the characteristics of three different tools: Microsoft Visual Basic 6.0 (for visual interface design), MathWorks Matlab 5.2 (for mathematical algorithms implementation) and Microsoft Access 97 (for data storage). The application uses Matlab files to store the functions that implement the algorithms (indexing methods, clustering techniques and image processing functions) while it uses Access tables to store the:

i) references to image files;
ii) declarations of the each indexing method;
iii) signatures associated to each image, computed using previously declared methods;
iv) declarations of the queries;
v) query results.

3.2 Using the application

The user interface allows an easy management of the interaction between methods and image archives, focusing user attention on the qualitative and quantitative analysis of querying strategy. We have also decided to enrich the interface with other functionalities that allow to simplify the management of the image archives allowing the creation of customized sets of images (*Image Tables*) or the grouping of existing *Tables* in *Image Catalogs*. *Tables* and *Catalogs* are used to set up the pattern and target sets for *Batch Queries*. *Batch queries* are useful to evaluate in a quantitative way a single indexing method or a more complex querying strategy.

The state of the image archives, i.e. their contents (images) and their applied indexing methods (indexes) can be browsed using a single form (**Figure 2**) that reports all the data in a single snapshot.

Once the user has chosen the image data sets that must be used as target or pattern sets, the user can step into the query definition phase.

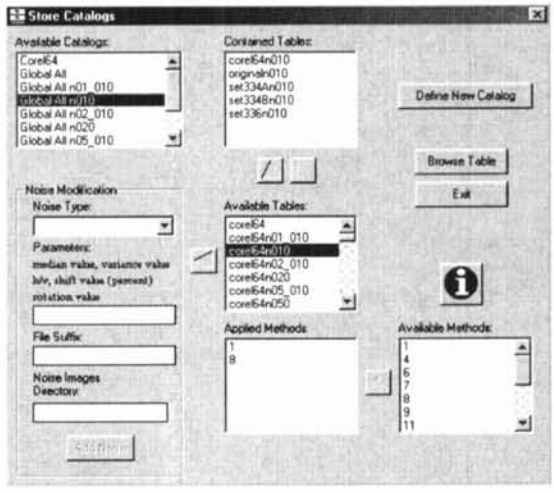

Figure 2. Catalogs management form

A single or batch query declaration, besides the choice of the pattern and target image sets, requires the choice of one (for single index) or more (for a composition of multiple indexes) indexing methods (see **Figure 3**). The choice of these methods involves the recall of their declarations and the typing in of their required parameters.

The system automatically stores the query definitions with their detailed results so as to allow a quantitative evaluation (see **Figure 4**); it allows a comparative analysis of different methods or different methods composition so to highlight their different retrieval properties. The next section shows how these features have been used for this purpose.

4 Selected experimental results

In this section we show how the framework depicted so far has been successfully used to reduce the computational costs of a single indexing method through calibrate querying strategy based on a chain of indexing methods. The methods used belong to a class of hierarchical correlation indexes [5] that exploit the wavelet transform.

Figure 5 shows the retrieval performances associated to two such methods. Roughly, their computational complexity results proportional to the number of

coefficients and to the number of planes involved in the indexing computation algorithm; so, method **B** results the most expensive indexing algorithm. The plots in **Figure 5** and **Figure 6** show the results of a *Batch Query* that uses 50 images, obtained from scanning operations, against an *Image Catalog* (170 indexed images) that contains the original version of the images. Queries have been run at a very coarse resolution level (i.e. 4x4 pixels). First each method has been applied alone (**Figure 5**), next we have used our environment to value possible strategies based on appropriate *sequences* based on the two methods (**Figure 6**). The two plots in **Figure 5** and **Figure 6** show the percentage of images correctly classified within the top scores listed in the column headings (*threshold for success*). Clearly, method **B** (**Figure 5**) results the best since it has the highest recall (number of positive matches).

The plot in **Figure 6** shows the results of two different instantiations of the sequence **AB**. The two cases differ in the percentage retained at the output of step **A** (20% and 30% respectively). The results show a slight improvement of the retrieval rate with a strong reduction of the retrieval computational costs with respect to the use of method **B** only (plot in **Figure 5)** whose computational cost is high.

The environment we have set up, allows to draw easily the following conclusions on this experiment:

- method **A**, being based on a very small signature, has a poor performance in terms of recall, but it is fairly precise; so it is a reliable filter to screen off false positives.
- a more complex methods (**B**) can re-arrange the ordering of the preliminary subset and is effective quite because of the pre-filtering done by method **A**.

5 Conclusions

We have presented a framework that supports the development and analysis of new indexing methods for content based image retrieval. It aims to standardize the operations done on indexes, allowing to combine heterogeneous indexing methods to produce an optimum indexing strategy. We have developed an application that implements these functionalities; it is currently used to support three research themes: evaluation of new indexing methods, development of indexing strategies and evaluation of clustering techniques.

6 References

[1] IBM Almaden Research Center "Query by Image And Video Content: The QBIC System". IEEE Computer, 1995, 9, 23-31.

[2] T. P. Minka, R.W. Picard, *Interactive learning using a society of models*. Technical Report N°. 249, MIT Media Laboratory, Cambridge, Mass, 1995.

[3] C. Carson, V. E. Ogle "Storage and retrieval of feature data for a very large online image collection". IEEE Computer Soc. Bull Techn. Comm. Data Eng., 1996, 19,(4).

[4] J. R. Smith, S. F. Chang "VisualSEEk: a fully automated content based image query system". ACM Multimedia, 1996, Nov.

[5] M. Albanesi, M. Ferretti, A. Giancane, "A Compact Wavelet Index for Retrieval in Image Database," Proc. 10th ICIAP, Sept. 27-29, Venezia, 1999, Comp. Society Press, pp. 927-931.

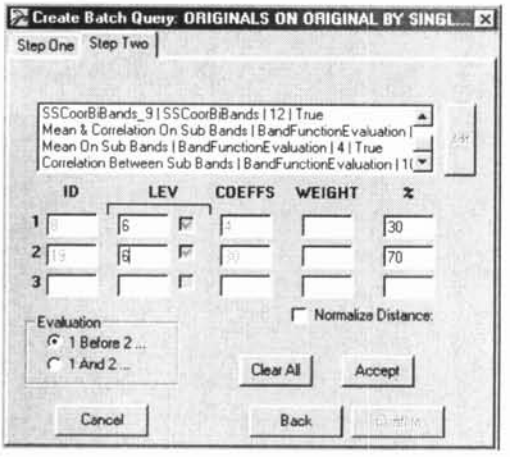

Figure 3. Methods composition in a batch query

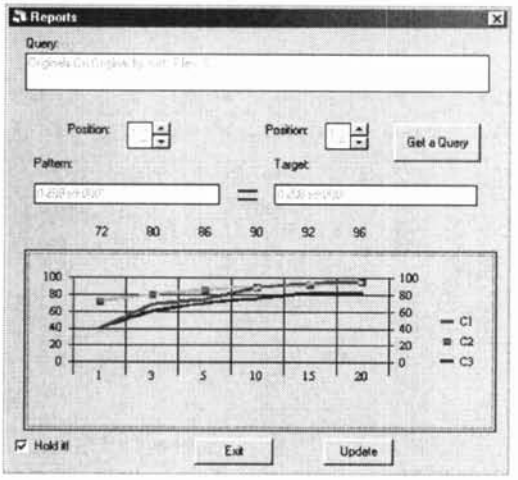

Figure 4. Comparative results analysis

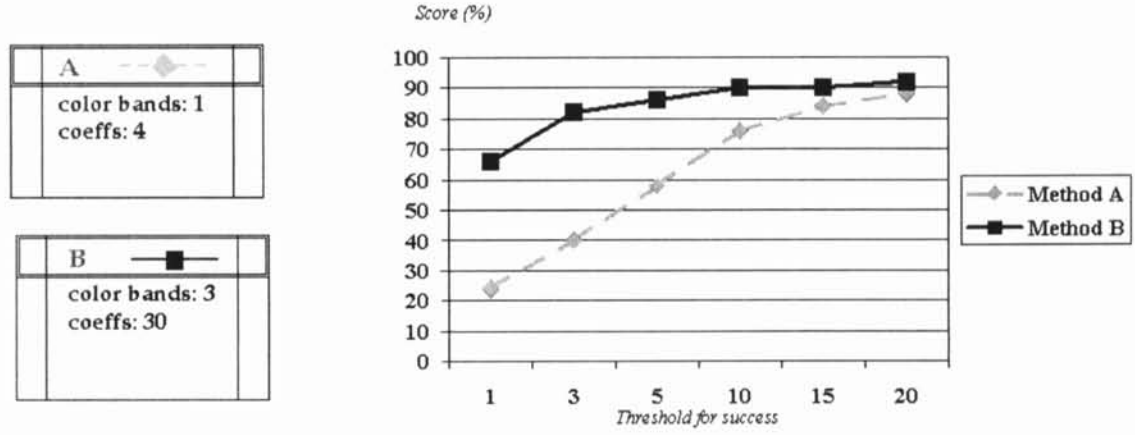

Figure 5. Experimental results - single indexes

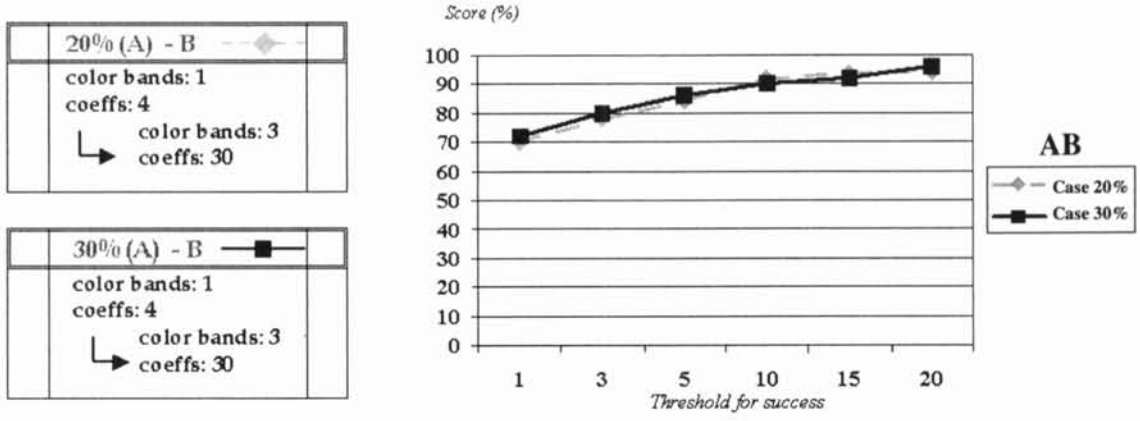

Figure 6. Experimental results - sequence of indexes